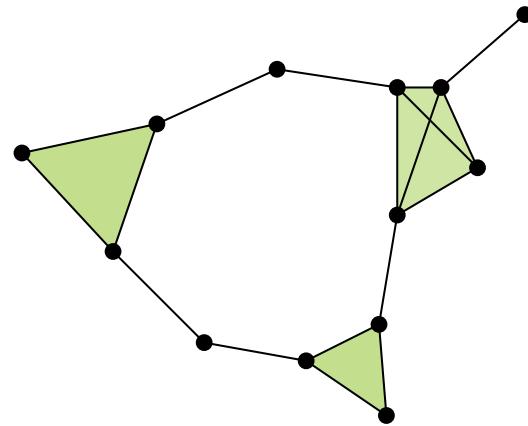


# Fast Construction of the Vietoris-Rips Complex



Afra Zomorodian  
Dartmouth Computer Science  
August 30, 2009

Sandia CAT – Santa Fe, NM

# Topological Data Analysis

- Input: scientific data (finite point set  $S \subseteq \mathbb{R}^d$ )
- Assumption: data was sampled from underlying space  $X$
- Goal: recover topology of  $X$

# Topological Data Analysis

- Input: scientific data (finite point set  $S \subseteq \mathbb{R}^d$ )
- Assumption: data was sampled from underlying space  $X$
- Goal: recover topology of  $X$
- Two step process
  1. Approximate  $X$  with a combinatorial representation,  
e.g. simplicial complex
  2. Compute topological invariant, e.g. (persistent) homology

# Topological Data Analysis

- Input: scientific data (finite point set  $S \subseteq \mathbb{R}^d$ )
- Assumption: data was sampled from underlying space  $X$
- Goal: recover topology of  $X$
- Two step process
  1. Approximate  $X$  with a combinatorial representation,  
e.g. simplicial complex
  2. Compute topological invariant, e.g. (persistent) homology
- Common misunderstanding: step 1 is easy, step 2 is hard

# Topological Data Analysis

- Input: scientific data (finite point set  $S \subseteq \mathbb{R}^d$ )
- Assumption: data was sampled from underlying space  $X$
- Goal: recover topology of  $X$
- Two step process
  1. Approximate  $X$  with a combinatorial representation,  
e.g. simplicial complex
  2. Compute topological invariant, e.g. (persistent) homology
- Common misunderstanding: step 1 is easy, step 2 is hard
- Reality: step 1 is hard, step 2 is easy (99% – 1%)

# Computing a Simplicial Complex

- Geometric methods
- Algebraic methods

# Computing a Simplicial Complex

- Geometric methods
  - alpha complex [Edelsbrunner *et al.* 83]
  - flow complex [Giesen & John 03]
  - good: relatively fast, small, embedded complexes
  - bad: still large, must compute Delaunay, hard past  $\mathbb{R}^3$
- Algebraic methods

# Computing a Simplicial Complex

- Geometric methods
  - alpha complex [Edelsbrunner *et al.* 83]
  - flow complex [Giesen & John 03]
  - good: relatively fast, small, embedded complexes
  - bad: still large, must compute Delaunay, hard past  $\mathbb{R}^3$
- Algebraic methods
  - Čech complex
  - Vietoris-Rips [Vietoris 27, Gromov 87]
  - good: simple
  - bad: slow, huge

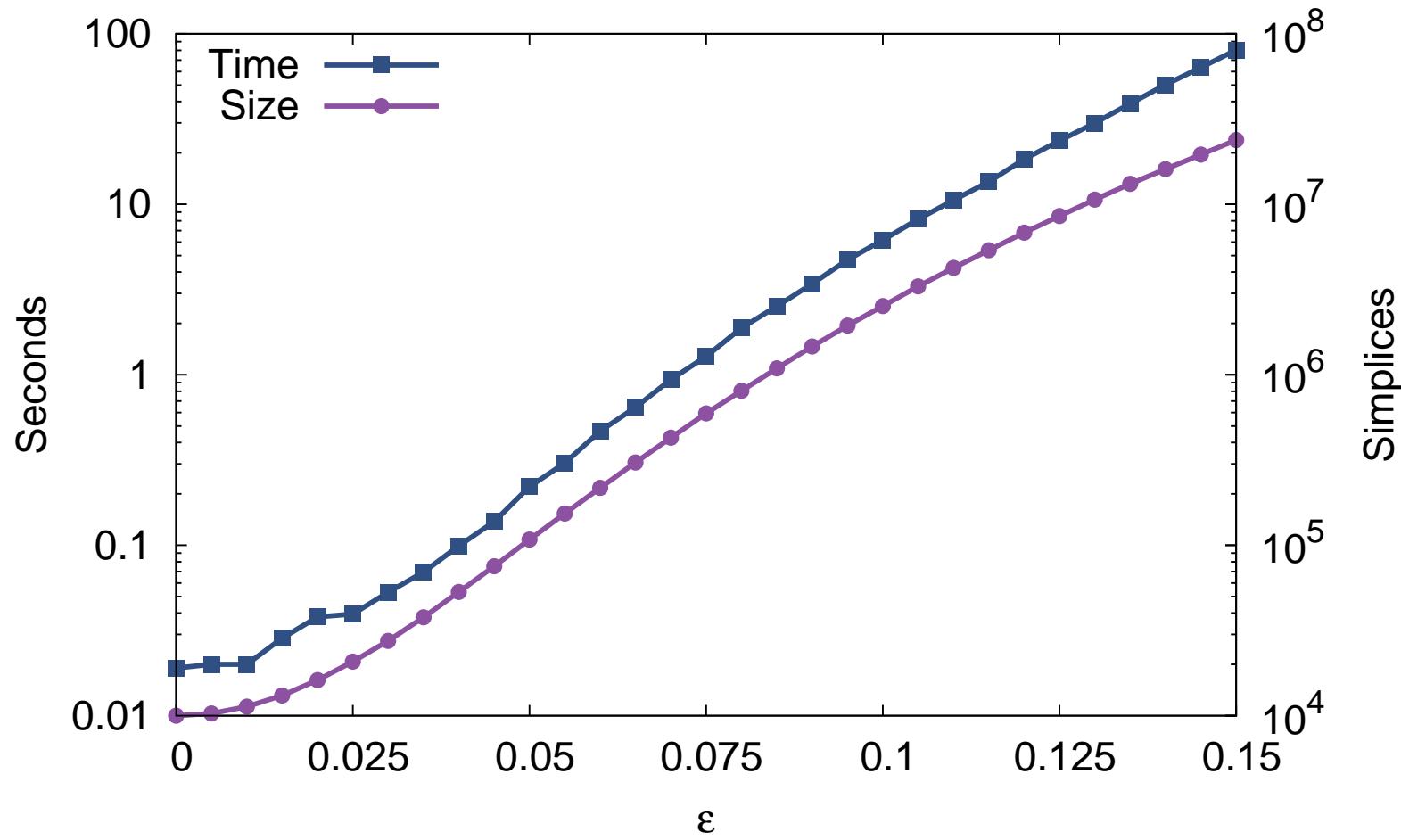
# Computing a Simplicial Complex

- Geometric methods
  - alpha complex [Edelsbrunner *et al.* 83]
  - flow complex [Giesen & John 03]
  - good: relatively fast, small, embedded complexes
  - bad: still large, must compute Delaunay, hard past  $\mathbb{R}^3$
- Algebraic methods
  - Čech complex
  - Vietoris-Rips [Vietoris 27, Gromov 87]
  - good: simple
  - bad: slow, huge
- Example: 2.5M to 50K to 50 landmarks [de Silva & Carlsson 04]

# Computing a Simplicial Complex

- Geometric methods
  - alpha complex [Edelsbrunner *et al.* 83]
  - flow complex [Giesen & John 03]
  - good: relatively fast, small, embedded complexes
  - bad: still large, must compute Delaunay, hard past  $\mathbb{R}^3$
- Algebraic methods
  - Čech complex
  - Vietoris-Rips [Vietoris 27, Gromov 87]
  - good: simple
  - bad: slow, huge
- Example: 2.5M to 50K to 50 landmarks [de Silva & Carlsson 04]
- My goal: make it faster

# How fast?



10,000 uniformly sampled points on unit 2-sphere in  $\mathbb{R}^3$   
1.1 million simplices in 2.5 seconds, 24 million in 80 seconds

# Background

- **simplicial complex**: a set  $K$  of finite sets s.t.  
if  $\sigma \in K$  and  $\tau \subseteq \sigma$ , then  $\tau \in K$ .
- If  $|\sigma| = k + 1$ ,  $\sigma$  is a  **$k$ -simplex** of **dimension  $k$** ,  $\dim(\sigma) = k$ .
- If  $\tau \subseteq \sigma \in K$ ,  $\tau$  is a **face** of  $\sigma$ , its **coface**.
- $\sigma \in K$  is **maximal** if it has no proper coface in  $K$ .
- $L \subseteq K$  is a **subcomplex** if it is a simplicial complex.
- **$k$ -skeleton of  $K$** :  $\{\sigma \in K \mid \dim(\sigma) \leq k\}$ .
- **filtration** is sequence of nested subcomplexes  
 $\emptyset = K_0 \subseteq K_1 \subseteq \dots \subseteq K_m = K$ . Then,  $K$  is a **filtered complex**.

# Vietoris-Rips Complex

- Given finite set  $S \subseteq \mathbb{R}^d$ , scale  $\epsilon \in \mathbb{R}$ .
  - Vietoris-Rips complex is  $\mathcal{V}_\epsilon(S) = \{\sigma \subseteq S \mid d(u, v) \leq \epsilon, \forall u \neq v \in \sigma\}$ , where  $d$  is the Euclidean metric.
  - That is, vertices of  $\sigma \in \mathcal{V}_\epsilon(S)$  are pairwise within distance  $\epsilon$
- 
- Sort simplices by  $\omega$  to get filtration ordering.
  - weight-filtered complex  $(K, f)$ ,  $f: K \rightarrow \mathbb{R}$ .

# Vietoris-Rips Complex

- Given finite set  $S \subseteq \mathbb{R}^d$ , scale  $\epsilon \in \mathbb{R}$ .
- Vietoris-Rips complex is  $\mathcal{V}_\epsilon(S) = \{\sigma \subseteq S \mid d(u, v) \leq \epsilon, \forall u \neq v \in \sigma\}$ , where  $d$  is the Euclidean metric.
- That is, vertices of  $\sigma \in \mathcal{V}_\epsilon(S)$  are pairwise within distance  $\epsilon$
- Maximum scale  $\hat{\epsilon}$ , weight function  $w: \mathcal{V}_{\hat{\epsilon}}(S) \rightarrow \mathbb{R}$ : If  $\sigma \in \mathcal{V}_{\hat{\epsilon}}(S)$ ,

$$w(\sigma) = \begin{cases} 0, & \dim(\sigma) \leq 0, \\ d(u, v), & \sigma = \{u, v\} \\ \max_{\tau \subset \sigma} w(\tau), & \text{otherwise.} \end{cases}$$

- Sort simplices by  $w$  to get filtration ordering.
- weight-filtered complex  $(K, f)$ ,  $f: K \rightarrow \mathbb{R}$ .

# Approach

1. neighborhood graph:  $(G, w)$ , undirected  $G = (V, E)$  and  $w: E \rightarrow \mathbb{R}$ .

# Approach

1. neighborhood graph:  $(G, w)$ , undirected  $G = (V, E)$  and  $w: E \rightarrow \mathbb{R}$ .

2. Given  $(G, w)$ , Vietoris-Rips expansion  $(\mathcal{V}(G), \omega)$

$$\mathcal{V}(G) = V \cup E \cup \left\{ \sigma \mid \binom{\sigma}{2} \in E \right\},$$

$$\omega(\sigma) = \begin{cases} 0, & \sigma \in V, \\ w(\{u, v\}), & \{u, v\} \in E, \\ \max_{\tau \subset \sigma} \omega(\tau), & \text{otherwise.} \end{cases}$$

# Approach

1. neighborhood graph:  $(G, w)$ , undirected  $G = (V, E)$  and  $w: E \rightarrow \mathbb{R}$ .
  - Vietoris-Rips neighborhood graph:  $(G_\epsilon(S), w)$ , where

$$V_\epsilon(S) = S,$$

$$E_\epsilon(S) = \{\{u, v\} \mid d(u, v) \leq \epsilon, u \neq v \in S\},$$

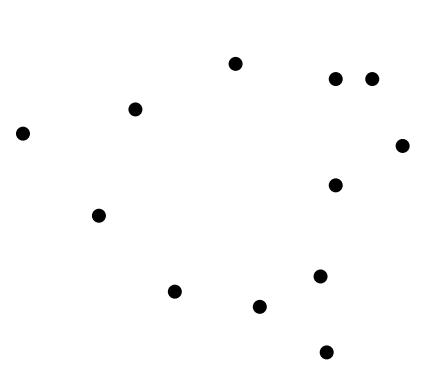
$$w(\{u, v\}) = d(u, v), \forall \{u, v\} \in E_\epsilon(S).$$

2. Given  $(G, w)$ , Vietoris-Rips expansion  $(\mathcal{V}(G), \omega)$

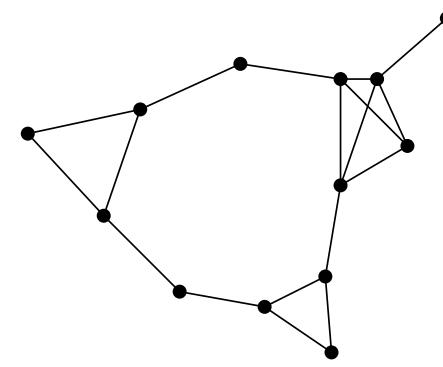
$$\mathcal{V}(G) = V \cup E \cup \left\{ \sigma \mid \binom{\sigma}{2} \in E \right\},$$

$$\omega(\sigma) = \begin{cases} 0, & \sigma \in V, \\ w(\{u, v\}), & \{u, v\} \in E, \\ \max_{\tau \subset \sigma} \omega(\tau), & \text{otherwise.} \end{cases}$$

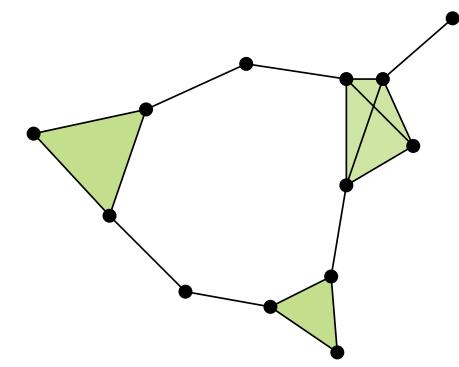
## Two Phases



(a) Points  $S$



(b)  $(G_{\hat{\epsilon}}(S), \omega)$



(c)  $(V_{\hat{\epsilon}}(S), \omega)$

- Input: (a)  $S \subseteq \mathbb{R}^d$ ,  $\hat{\epsilon} \in \mathbb{R}$ , and  $k \in \mathbb{Z}^{>1}$ 
  - Geometric Phase: Compute (b)  $(G_{\hat{\epsilon}}(S), \omega)$
  - Topological Phase: Compute (c)  $k$ -skeleton of  $(V(G_{\hat{\epsilon}}(S)), \omega)$
- $V_{\hat{\epsilon}}(S) = V(G_{\hat{\epsilon}}(S))$

# Computing the Neighborhood Graph

- Classic family of nearest or near neighbors problems
- Exact  $\epsilon$ -near and  $(1 + \delta)$ -approximate
  - all-pairs: brute-force  $O(n^2)$
  - scanning: project onto random axis
  - Kd-tree + search [Friedman *et al.* 77]
  - BBD-tree + priority search [Arya *et al.* 98]
- Randomized  $(1 - \gamma)$ : locality sensitive hashing [Andoni & Indyk 08]

# Computing the Neighborhood Graph

- Classic family of nearest or near neighbors problems
- Exact  $\epsilon$ -near and  $(1 + \delta)$ -approximate
  - all-pairs: brute-force  $O(n^2)$
  - scanning: project onto random axis
  - Kd-tree + search [Friedman *et al.* 77]
  - BBD-tree + priority search [Arya *et al.* 98]
- Randomized  $(1 - \gamma)$ : locality sensitive hashing [Andoni & Indyk 08]
- Landmarked [de Silva & Carlsson 04]
  - landmarks  $L \subseteq S$ , potential witnesses  $S - L$ .
  - $\epsilon$ -witness neighborhood graph  $G_{\epsilon, L}(S) = (L, E_{\epsilon, L})$ 
$$E_{\epsilon, L} = \{\{u, v\} \mid \max(d(u, w), d(v, w)) \leq \epsilon, u, v \in L, w \in S - L\},$$

# Inductive Algorithm

- Input:  $(G, w)$ ,  $k \in \mathbb{Z}^{>1}$
- Invariant:  $i$ -skeleton at top of the loop

# Inductive Algorithm

- Input:  $(G, w)$ ,  $k \in \mathbb{Z}^{>1}$
- Invariant:  $i$ -skeleton at top of the loop

INDUCTIVE-VR( $G, k$ )

- 1     $\mathcal{V} = G.V \cup G.E$
- 2    **for**  $i = 1$  **to**  $k$
- 3        **foreach**  $i$ -simplex  $\tau \in \mathcal{V}$
- 4             $N = \bigcap_{u \in \tau} \text{LOWER-NEIGHBORS}(G, u)$
- 5            **foreach**  $v \in N$
- 6                 $\mathcal{V} = \mathcal{V} \cup \{\tau \cup \{v\}\}$
- 7    **return**  $\mathcal{V}$

# Inductive Algorithm

- Input:  $(G, w)$ ,  $k \in \mathbb{Z}^{>1}$
- Invariant:  $i$ -skeleton at top of the loop

INDUCTIVE-VR( $G, k$ )

- 1  $\mathcal{V} = G.V \cup G.E$
- 2 **for**  $i = 1$  **to**  $k$
- 3     **foreach**  $i$ -simplex  $\tau \in \mathcal{V}$
- 4          $N = \bigcap_{u \in \tau} \text{LOWER-NEIGHBORS}(G, u)$
- 5         **foreach**  $v \in N$
- 6              $\mathcal{V} = \mathcal{V} \cup \{\tau \cup \{v\}\}$
- 7 **return**  $\mathcal{V}$

LOWER-NEIGHBORS( $G, u$ )

- 1 **return**  $\{v \in G.V \mid u > v, \{u, v\} \in G.E\}$

# Incremental Algorithm

- Place total order on  $S$
- Invariant:  $v$  is added with all cofaces that have  $v$  as maximal vertex

# Incremental Algorithm

- Place total order on  $S$
- Invariant:  $v$  is added with all cofaces that have  $v$  as maximal vertex

INCREMENTAL-VR( $G, k$ )

- 1  $\mathcal{V} = \emptyset$
- 2 **foreach**  $u \in G.V$
- 3      $N = \text{LOWER-NEIGHBORS}(G, u)$
- 4     ADD-COFACES( $G, \mathcal{V}, \{v\}, N, k$ )
- 5 **return**  $\mathcal{V}$

## Incremental Algorithm II

- ADD-COFACES( $G, \mathcal{V}, \tau, N, k$ ) invariants:
  1.  $\tau$  should be in  $\mathcal{V}$
  2.  $N$  is the set of  $\tau$ 's lower neighbors

# Incremental Algorithm II

- ADD-COFACES( $G, \mathcal{V}, \tau, N, k$ ) invariants:
  1.  $\tau$  should be in  $\mathcal{V}$
  2.  $N$  is the set of  $\tau$ 's lower neighbors

ADD-COFACES( $G, \mathcal{V}, \tau, N, k$ )

```
1   $\mathcal{V} = \mathcal{V} \cup \{\tau\}$ 
2  if  $\dim(\tau) \geq k$ 
3      return
4  else
5      foreach  $v \in N$ 
6           $\sigma = \tau \cup \{v\}$ 
7           $M = N \cap \text{LOWER-NEIGHBORS}(G, v)$ 
8          ADD-COFACES( $G, \mathcal{V}, \sigma, M, k$ )
```

# Maximal Algorithm

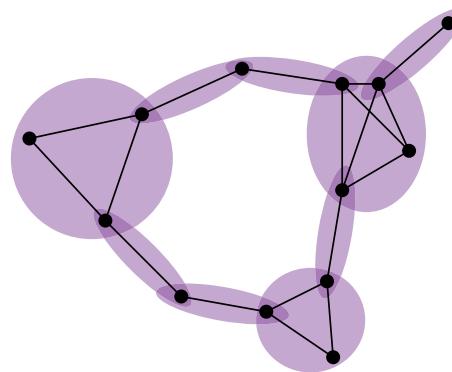
- maximal clique: maximal complete subgraph

# Maximal Algorithm

- maximal clique: maximal complete subgraph
- Idea: maximal cliques in  $G$  are maximal simplices in  $\mathcal{V}$

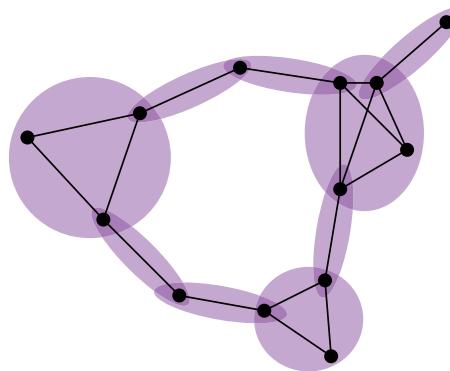
# Maximal Algorithm

- maximal clique: maximal complete subgraph
- Idea: maximal cliques in  $G$  are maximal simplices in  $\mathcal{V}$



# Maximal Algorithm

- maximal clique: maximal complete subgraph
- Idea: maximal cliques in  $G$  are maximal simplices in  $\mathcal{V}$

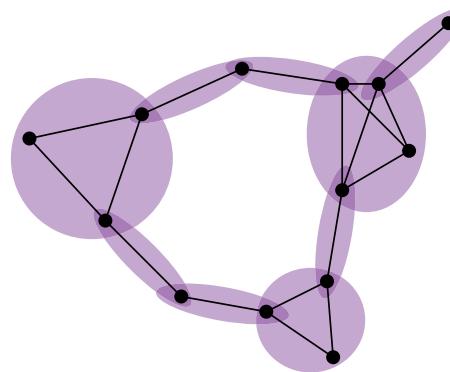


MAXIMAL-VR( $G, k$ )

- 1  $C = \text{IK-GX}(G)$
- 2  $\mathcal{V} = \text{GENERATE-COMBINATIONS}(C, k)$
- 3 **return**  $\mathcal{V}$

# Maximal Algorithm

- maximal clique: maximal complete subgraph
- Idea: maximal cliques in  $G$  are maximal simplices in  $\mathcal{V}$



MAXIMAL-VR( $G, k$ )

- 1  $C = \text{IK-GX}(G)$
- 2  $\mathcal{V} = \text{GENERATE-COMBINATIONS}(C, k)$
- 3 **return**  $\mathcal{V}$

- IK-GX

[Bron & Kerbosch 73, Koch 01, Cazals & Karande 05]

# Computing the Weight Function

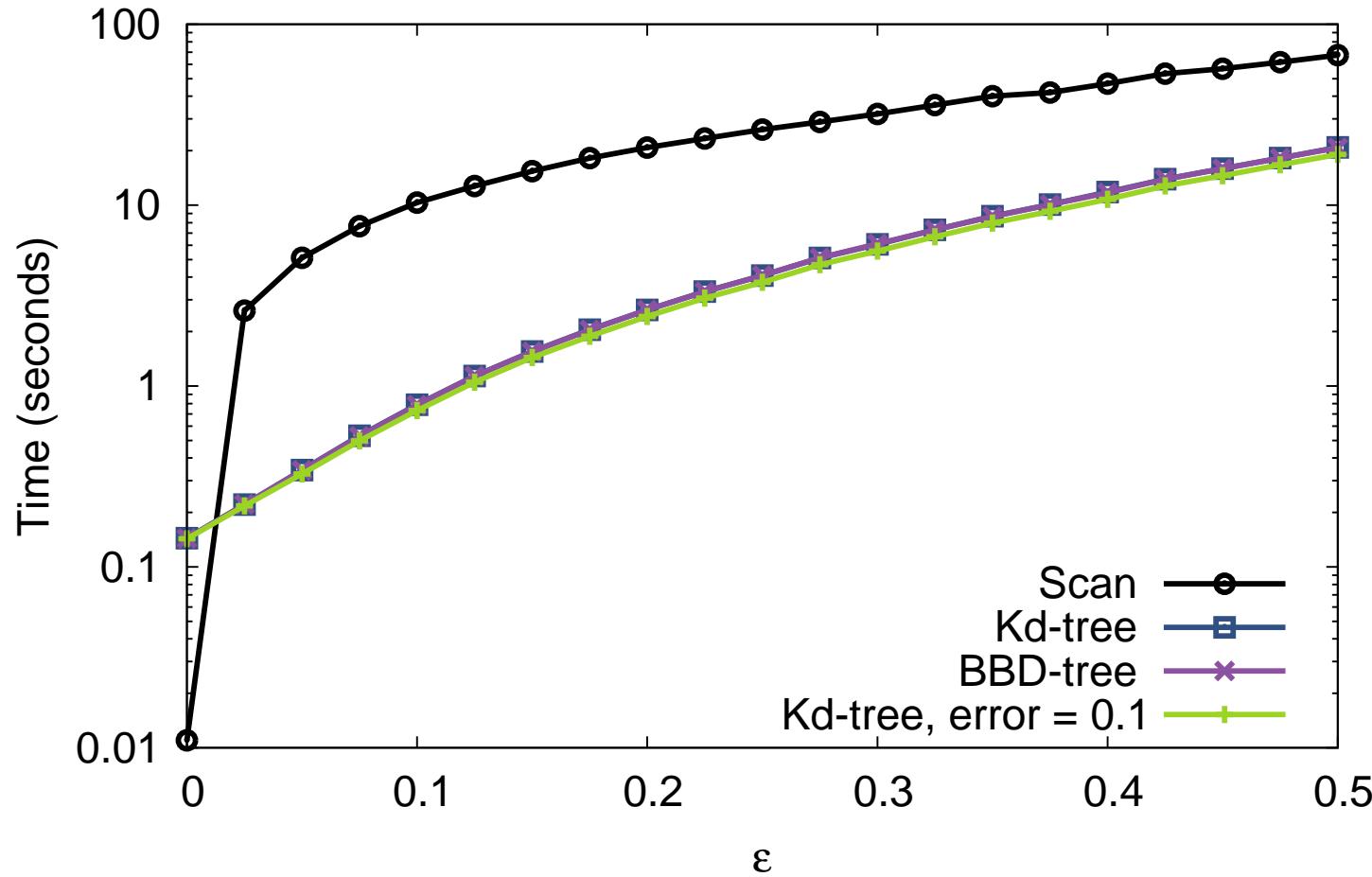
COMPUTE-WEIGHT-FUNCTION( $\mathcal{V}, w$ )

- 1 **foreach** vertex  $v \in \mathcal{V}$
- 2      $\omega(v) = 0$
- 3 **foreach** edge  $e \in \mathcal{V}$
- 4      $\omega(e) = w(e)$
- 5 **foreach** simplex  $\sigma \in \mathcal{V}$
- 6     COMPUTE-WEIGHT( $\sigma, \omega$ )
- 7 **return**  $\omega$

COMPUTE-WEIGHT( $\sigma, \omega$ )

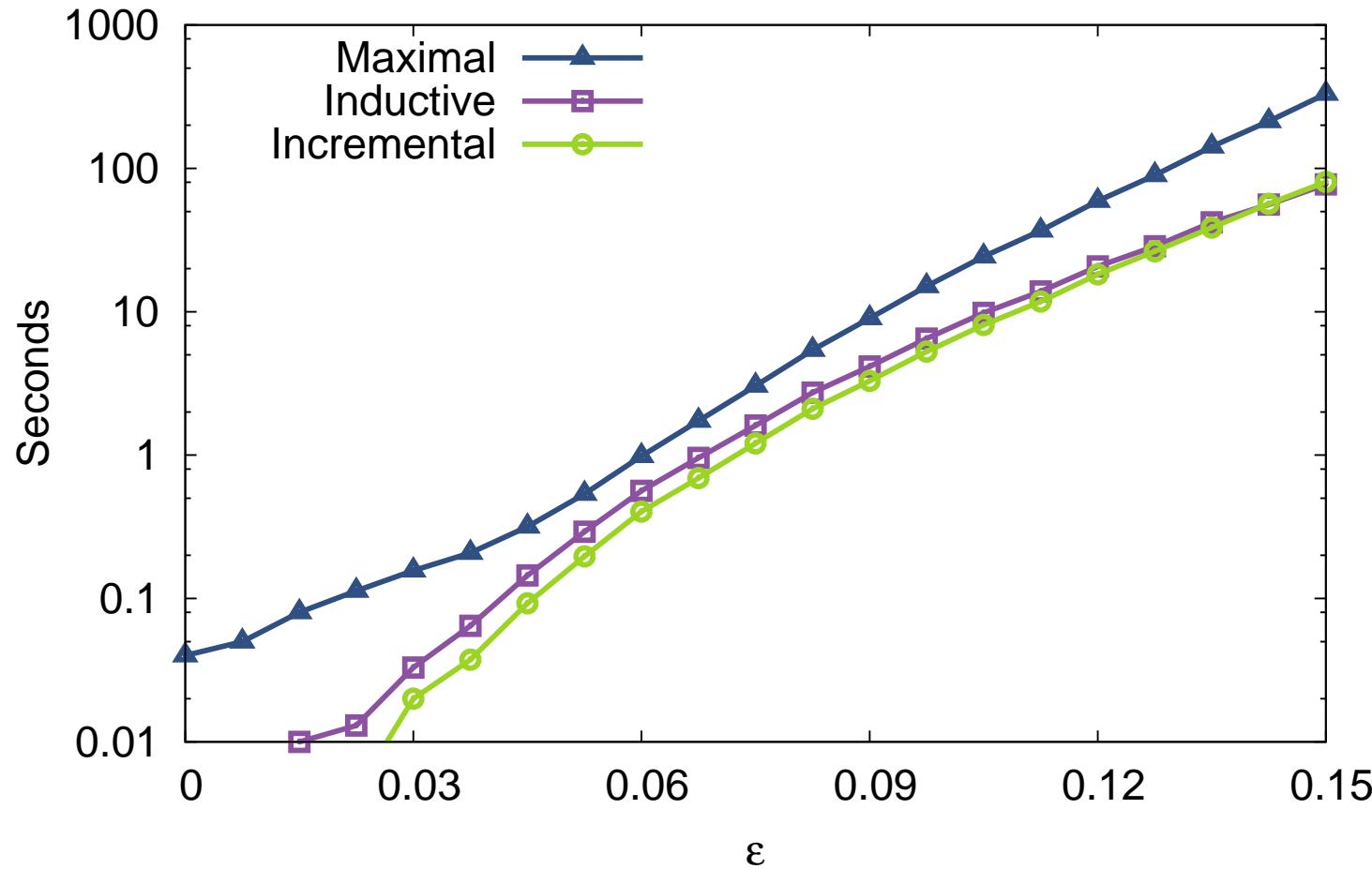
- 1 **if**  $\omega(\sigma)$  is defined
- 2     **return**  $\omega(\sigma)$
- 3 **else**
- 4     **return**  $\omega(\sigma) = \max_{\tau \subset \sigma} \text{COMPUTE-WEIGHT}(\tau, \omega)$

## Phase I



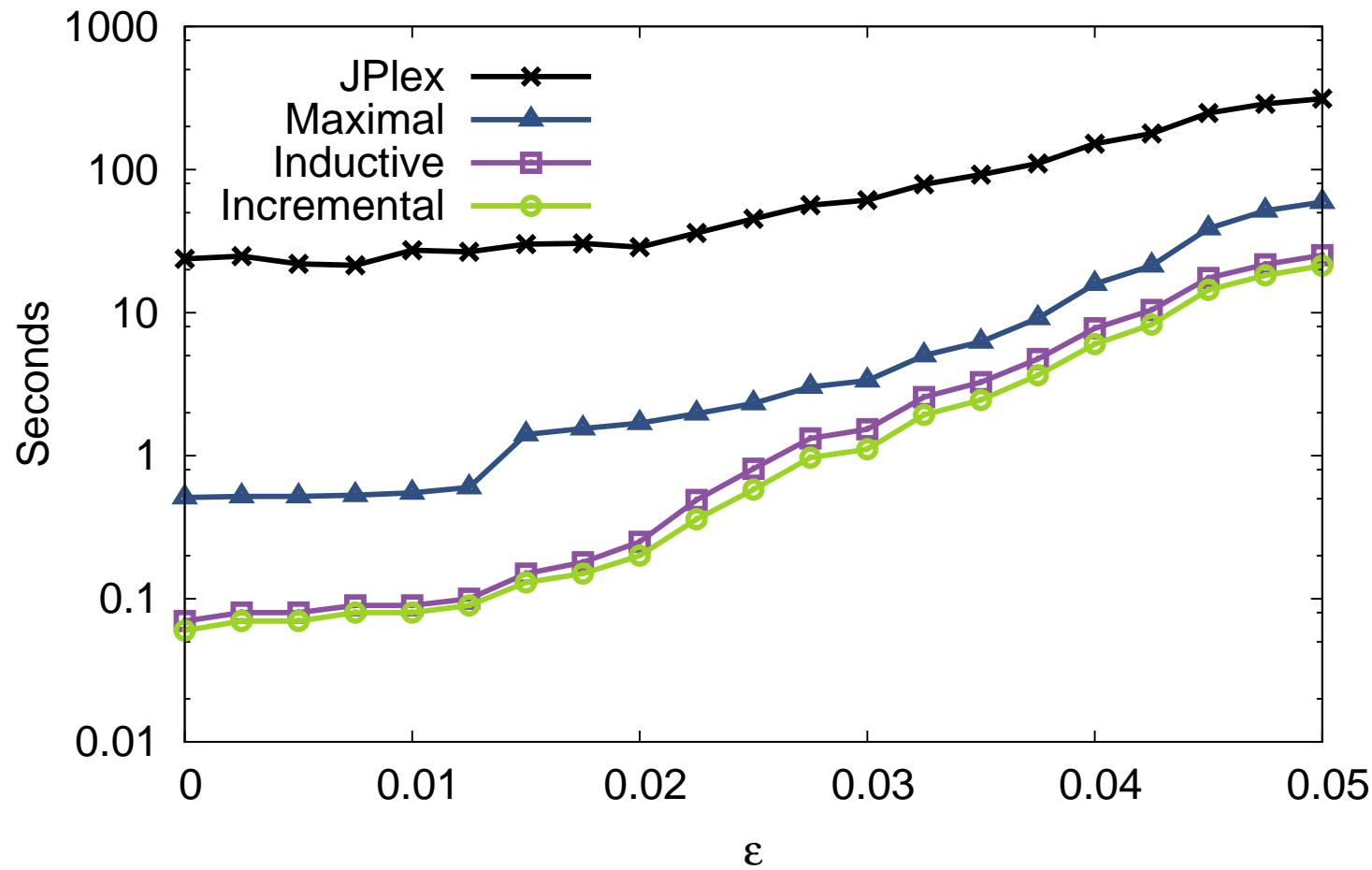
50,000 uniformly sampled points from a unit 3-sphere in  $\mathbb{R}^8$ .  
11.6 million edges in less than 20 seconds – E<sup>2</sup>LSH not viable

## Phase II



10,000 uniformly sampled points from a unit 2-sphere embedded in  $\mathbb{R}^3$ .  
24 million simplices in 80 seconds

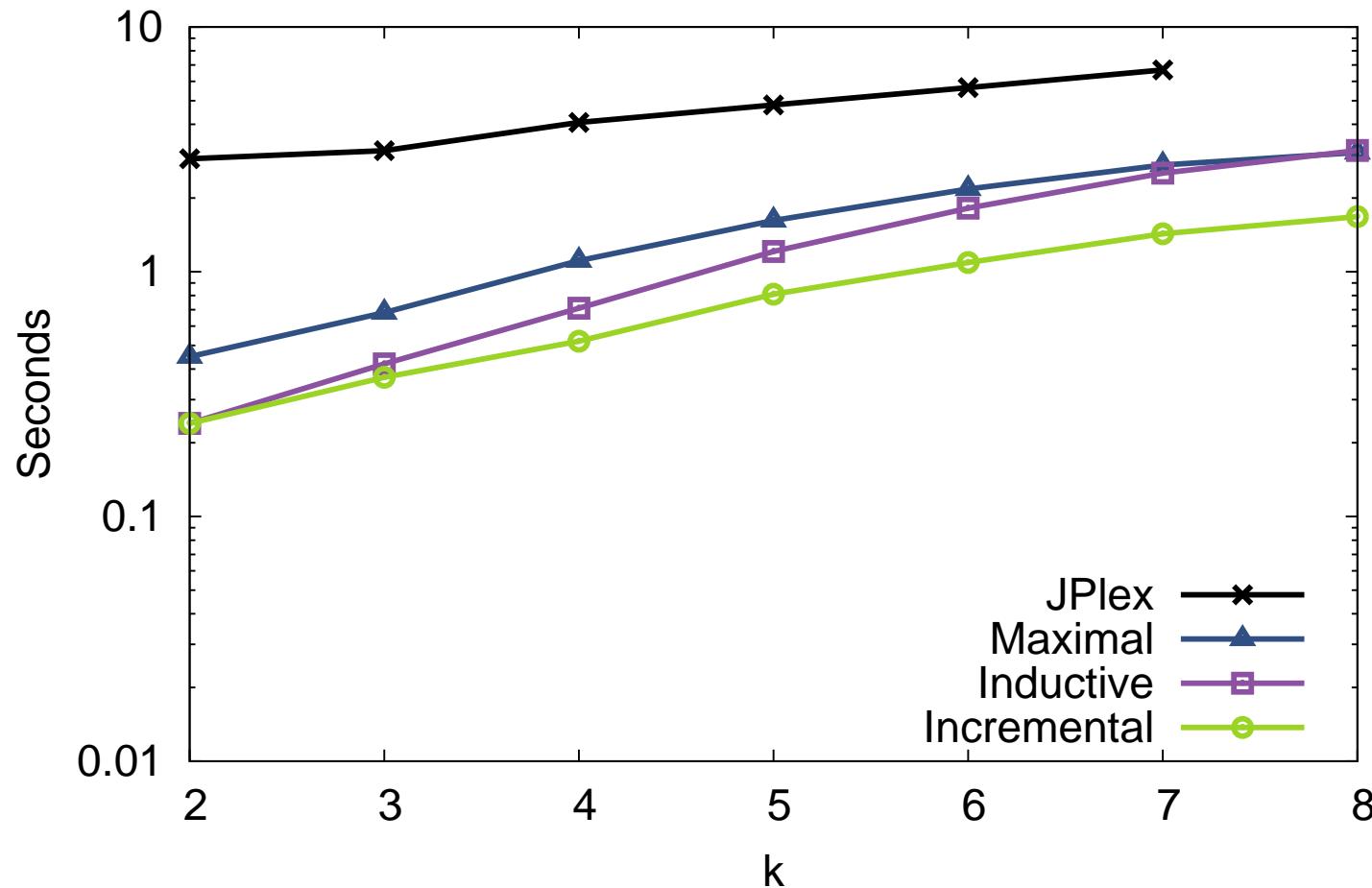
# Full Construction



Stanford *bunny*: 34,837 points in  $\mathbb{R}^3$

9,714,912 simplices in 21.27 seconds (JPlex 313.38 seconds)

# Higher Dimensions



*natural*: 10,000 points in  $\mathbb{R}^8$  (van Hateren-Mumford  $k = 30$ , cut = 20)  
539,627 simplices in 1.68 seconds (JPlex 6.68 seconds for  $k = 7$ )

# Conclusion

- VR complex is beautiful as it separates geometry from topology
- Exact nearest neighbors are fast enough for our techniques
- Three algorithms
  - INDUCTIVE-VR is intuitive
  - INCREMENTAL-VR is natural for filtered data [Collins *et al.* 04]
  - MAXIMAL-VR gives minimum description in maximal simplices

# Conclusion

- VR complex is beautiful as it separates geometry from topology
- Exact nearest neighbors are fast enough for our techniques
- Three algorithms
  - INDUCTIVE-VR is intuitive
  - INCREMENTAL-VR is natural for filtered data [Collins *et al.* 04]
  - MAXIMAL-VR gives minimum description in maximal simplices
- Faster (general and generic, too!)
- Both steps of MAXIMAL-VR are parallelizable

# Conclusion

- VR complex is beautiful as it separates geometry from topology
- Exact nearest neighbors are fast enough for our techniques
- Three algorithms
  - INDUCTIVE-VR is intuitive
  - INCREMENTAL-VR is natural for filtered data [Collins *et al.* 04]
  - MAXIMAL-VR gives minimum description in maximal simplices
- Faster (general and generic, too!)
- Both steps of MAXIMAL-VR are parallelizable
- *But step 2 remains: compute homology*

## Current Project

- *bunny*: 34,837 points at  $\epsilon = 0.05$ , 3-dimensional

Generation: 20.82

Homology: 63.88

Total: 84.70

Size: 9,714,912

## Current Project

- *bunny*: 34,837 points at  $\epsilon = 0.05$ , 3-dimensional

Generation: 20.82

Homology: 63.88

Total: 84.70

Size: 9,714,912

Collapsed: 39.63

Size: 1,303

## Current Project

- *natural*: 10,000 points at  $\epsilon = 0.11$ , 8-dimensional

Generation: 1.68

Homology: 4.93

Total: 6.61

Size: 539,627

## Current Project

- *natural*: 10,000 points at  $\epsilon = 0.11$ , 8-dimensional

Generation: 1.68

Homology: 4.93

Total: 6.61

Size: 539,627

Collapsed: 0.60

Size: 7,531

## Current Project

- *natural*: 10,000 points at  $\epsilon = 0.11$ , 8-dimensional

Generation: 1.68

Homology: 4.93

Total: 6.61

Size: 539,627

Collapsed: 0.60

Size: 7,531

- Not filtered yet.

# Acknowledgments

- STL, Boost, BGL [Siek *et al.* 02]
- ANN [Mount & Arya 06]
- E<sup>2</sup>LSH [Andoni 09]
- Gunnar Carlsson for early discussions
- Frédéric Cazals for maximal clique survey
- Harlan Sexton for JPlex
- Henry Adams for JPlex tutorial
- Funding agencies:

DARPA HR 0011-06-1-0038

ONR N 00014-08-1-0908

NSF CCF 0845716